Connexité dans l'urgence[†]

Cyril Gavoille[‡], Pierre Halftermeyer[†§]

LaBRI, Université de Bordeaux, France

Nous présentons une structure de données permettant de répondre rapidement aux requêtes de connexité dans un réseau en présence d'un nombre arbitraire de sommets ou de liens défaillant. Plus précisément, après le pré-calcul d'un graphe G, on peut déterminer si u et v sont connectés dans le graphe $G \setminus X$ pour toute paire de sommets u, v et tout sous-ensemble X de sommets ou d'arêtes de G. Le temps de réponse ne dépend que de $|X|$ et du genre de G. La structure de données d'espace $O(gn)$ peut être distribuée en n étiquettes de $O(g \log n)$ bits.

Mots-clefs : Planification de l'urgence, connexité, schéma d'étiquetage, ensembles interdits, graphes de genre borné

1 Introduction

Motivations. Une question naturelle est de savoir comment la suppression d'un lien ou d'un nœud peut modifier la connectivité d'un réseau modélisé par un graphe. Les concepts de points d'articulation et d'isthmes permettent de répondent précisément à cette question.

Déterminer rapidement si u et v sont toujours connectés dans $G \setminus X$ lorsque X contient plus de deux sommets est beaucoup plus difficile, même après un pré-calcul de G. Si X ne contient pas de sommets, mais seulement des arêtes, le problème peut être résolu [7] en temps $O(|X| \log^2 n \log\log n)$ où n est le nombre de sommets de G. Cependant, aucune solution meilleure que l'application du plus rapide des algorithmes dynamiques n'est connue dès que X peut contenir plusieurs sommets. La complexité s'écroule alors en $O(|X|\sqrt{n})$ [5].

La motivation pour l'étude de ce problème est la gestion d'une situation de crises lors, par exemple, d'une attaque parallèle sur des liens ou des nœuds d'un réseau. Après une telle attaque, on souhaite déterminer si deux sommets sont encore connectés par un chemin du graphe, et ce d'autant plus rapidement qu'il est impératif de raccourcir la période d'instabilité post-attaque. Dit autrement, après une attaque il devient urgent de comprendre la nouvelle connectivité du réseau sans avoir à reparcourir le graphe en entier.

D'un point de vue technique, il est tout à fait envisageable de faire un pré-calcul polynomial en la taille de G si cela permet de répondre aux requêtes de connexions lors de la crise en temps poly-logarithmique.

Comme identifiées par Thorup *et al.* [7] des situations comme la destruction partiel d'un réseau routier après un tremblement de terre, ou la dissémination malicieuse de virus informatiques, impliquent des modifications topologiques simultanées en nombre qui doivent être gérées le plus rapidement possible. Un algorithme même linéaire en n n'est d'aucune utilité dans ce cas, puisqu'on souhaite réagir en un temps essentiellement dépendant de la taille de l'attaque (ou du désordre), pas de celle du graphe ! On remarque que ce paradigme de *planification de l'urgence* n'a que faire d'une solution qui serait efficace seulement en moyenne (en temps amorti), puisque la perspective de prendre beaucoup de temps pour quelques modifications est inacceptable dans une situation d'urgence.

Travaux existant. Le maintien d'une structure de données dynamique pour la connexité dans un graphe est un problème fondamental en algorithmique notablement connu pour sa difficulté. Le meilleur algorithme garanti une complexité de $O(\sqrt{n})$ par mise à jour, alors que la borne inférieure est seulement de $\Omega(\log n)$ [5]. Des solutions en temps poly-logarithmique existent mais soit il s'agit de temps amorti (et

[†]Supported by the ANR project "DISPLEXITY", the European Project "EULER" and the équipe-projet INRIA "CÉPAGE".
[‡]Membre de l'Institut Universitaire de France. gavoille@labri.fr
[§]pierre.halftermeyer@labri.fr

ne peuvent s'appliquer à notre problème), soit l'ensemble X ne doit pas contenir de nœuds. Des solutions poly-logarithmiques existent cependant dans le cas d'un graphe planaire [6], mais soit les suppressions de sommets ne sont pas autorisées, soit le plongement du graphe doit être préservé par l'ajout de sommets ou d'arêtes. Comme observé par [7], la résolution de ce problème de connectivité avec planification est une étape préliminaire à une solution poly-logarithmique complètement dynamique.

Courcelle *et al.* [4] ont montré que les graphes de *clique-width* k ont une structure de données d'espace $O(k^2 n)$ permettant de déterminer en temps $O(k^2 \log n)$ si u, v sont connectés dans $G \setminus X$. En agrandissant la taille de la structure d'un facteur $\log n$, de surcroît, la distance et la plus courte route entre u, v dans $G \setminus X$ peuvent être déterminées. Dans le cas planaire, k est non-borné et une solution *ad-hoc* de taille $O(n)$ existe pour la connectivité [3] avec un temps de requête de $O(|X| \log n)$. Les graphes de dimensions doublante bornées ont été investis par Abraham *et al.* [1] pour la connexité et le routage de presque plus court chemins.

Structure de données distribuées. En fait les structures de données de taille $O(\lambda n)$ développées par [1, 3, 4] peuvent être découpées en n étiquettes de taille $O(\lambda)$. Plus précisément, chaque sommet u de G reçoit une étiquette notée $L(u,G)$ de sorte qu'une requête (u,v,X) peut être résolue en calculant $A(L(u,G), L(v,G), \{L(w,G) : w \in X\})$, c'est-à-dire en combinant par un algorithme donné A les étiquettes des sommets impliqués dans la requête[¶]. La paire de fonctions $\mathcal{L} = \langle L, A \rangle$ forme un *schéma d'étiquetage* pour la classe de graphes considérée.

La possibilité d'être distribuées rend ces solutions particulièrement adaptées au routage par exemple. Étant donné un ensemble X de liens en pannes, de routeurs défectueux ou non désirables, un nœud u pourra déterminer le *next-hop* vers une destination v sur la base de sa propre table de routage et celle du correctif ou *patch* pour X – qui sont les étiquettes. La taille de ce *patch* est $O(\lambda |X|)$ ce qui est indépendant de la taille du réseau. Notons enfin que ce modèle autorise chaque routeur u de choisir un ensemble privé X_u de routeurs interdits (ensemble éventuellement variable au court du temps) ce qui capture de manière réaliste[‖] les *policy routing* de BGP.

2 Notre contribution

Dans ce cadre nous étendons le résultat de [3] pour les graphes planaires à tout graphe plongé sur une surface (orientable ou non). Dans la suite G est un graphe à n sommets de genre Eulérien g, qui est le minimum entre le genre non-orientable et le double du genre orientable de G. On notera $\textsc{Conn}(u,v,X)$ le prédicat indiquant si les sommets u et v sont dans la même composante connexe du graphe $G \setminus X$.

Théorème 1 *La classe des graphes à n sommets de genre Eulérien g admet un schéma d'étiquetage pour les requêtes de connexité avec obstacle produisant des étiquettes de $O(g \log n)$ bits. Les n étiquettes de chaque graphe peuvent être calculées en temps $O((n+g)(g+\log n))$ étant donné un plongement de genre Eulérien g du graphe. Le temps de réponse à une requête $\textsc{Conn}(u,v,X)$ est $O(g|X| \cdot \log(g|X|))$.*

Notre solution prend appui sur le cas planaire ($g = 0$) qui a été résolu par Courcelle *et al.* [3]. On notera $\mathcal{L}_0 = \langle L_0, A_0 \rangle$ ce schéma qui sera utilisé comme *boîte noire* par notre schéma. L'esquisse de la preuve du théorème 1 repose sur la combinaison des lemmes 1, 2 et 3 présentés ci-dessous.

Étiquetage. L'étiquetage est basé sur la construction d'un *schéma polygonal canonique* noté $(\hat{G}, \mathcal{P})$. Il peut être obtenu à partir de la carte combinatoire de G en temps $O((n+g)(g+\log n))$ d'après [8]. Le schéma polygonal $(\hat{G}, \mathcal{P})$ est défini comme suit :

- $\hat{G}$ est une carte planaire de $O(gn)$ sommets.
- Le bord $\mathcal{B}$ de sa face extérieure de $\hat{G}$ est la réunion de $2g$ chemins $P_1, \ldots, P_{2g}$ qui forment les côtés du polygone $\mathcal{P}$. Les chemins sont deux à deux distincts sauf les sommets-extrémités des côtés consécutifs.
- On peut retrouver la carte combinatoire de G en identifiant deux à deux les P_i selon un couplage et un sens prédéfinis canoniquement ne dépendant que de g et de son orientabilité (voir fig. 1).

¶. Les arêtes ab potentiellement dans X sont étiquetées par la paire $\langle L(a,G), L(b,G) \rangle$.

‖. Bien sûr une mauvaise distribution des ensembles interdits privés X_{u_i} peut rendre le routage de u à v impossible, ce qui n'est pas le cas si tous les ensembles sont coordonnés (et si bien sûr u et v sont connectés dans $G \setminus X$).

Chaque sommet u de G se trouve représenté par un ou plusieurs sommets $\hat{u}_1, \hat{u}_2, \ldots$ de $\hat{G}$. Les sommets de G représentés par un sommet de $\mathcal{B}$ ont au plus $2g$ représentants dans $\hat{G}$, alors que les sommets représentés par un sommet qui n'est pas dans $\mathcal{B}$ ont un seul représentant.

On applique sur le graphe $\hat{G}$ le schéma d'étiquetage $\mathcal{L}_0$ dédié aux graphes planaires.

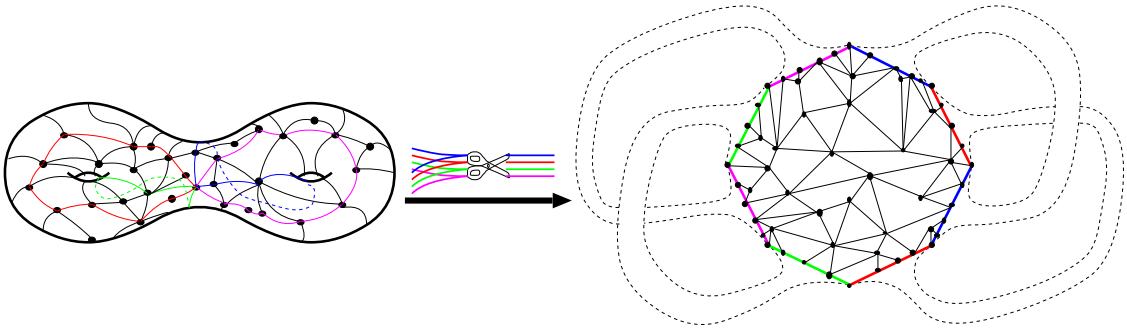

FIGURE 1: Graphe G plongé sur un double tore (de genre orientable 2 mais de genre Eulérien 4) et un schéma polygonal canonique $(\hat{G}, \mathcal{P})$.

On définit $\textsc{Id} : \mathcal{B} \to \{1, \ldots, |\mathcal{B}|\}$ la fonction qui donne l'indice d'apparition d'un sommet dans le bord (l'indice 1 étant l'indice du sommet commun à P_{2g} et P_1) et $\textsc{Next} : \mathcal{B} \to \mathcal{B}$ la fonction qui donne le sommet suivant dans $\mathcal{B}$ dans le sens anti-trigonométrique.

L'étiquetage des sommets de G est relativement élémentaire, la difficulté du schéma résidant dans le traitement de la requête. L'étiquette d'un sommet u de G est composée :

- de la configuration du schéma polygonal, à savoir son genre g et son orientabilité (soit $O(\log g)$ bits) et la liste des indices $\textsc{Id}$ des sommets-extrémités du polygone $\mathcal{P}$ (soit $O(g \log n)$ bits) ;
- et, pour chaque représentant $\hat{u}_i$ de u, du triplet (μ, j, μ') où $\mu = L_0(\hat{u}_i, \hat{G})$, $j = \textsc{Id}(\hat{u}_i)$ et $\mu' = L_0(\textsc{Next}(\hat{u}_i), \hat{G})$ avec la convention $j = \mu' = \perp$ si $\hat{u}_i \notin \mathcal{B}$.

Lemme 1 *L'étiquetage des sommets de G peut se faire en temps $O(gn)$. La longueur d'une étiquette est $O(g \log n + g \cdot \ell_0(gn))$ bits où $\ell_0(gn)$ est la longueur maximum d'une étiquette d'un sommet de $\hat{G}$ construite par le schéma $\mathcal{L}_0$. Les étiquettes des sommets de G sont d'au plus $O(g \log n)$ bits d'après [3].*

Traitement d'une requête. Chaque requête $\textsc{Conn}(u, v, X)$ pour le graphe G est traitée en construisant un graphe auxiliaire G^+, fonction de (u, v, X), contenant $O(g \cdot |X|)$ sommets et arêtes. Il est produit à partir des étiquettes des sommets impliqués dans la requête. La réponse finale à la requête est entièrement déterminée par le fait que deux sommets spéciaux $\tilde{u}$ et $\tilde{v}$ de G^+ sont connectés ou non dans G^+ (cf. lemme 3).

Plus précisément, le graphe G^+ (voir fig. 2) est défini comme suit, où $\hat{X}$ est l'ensemble des sommets de $\hat{G}$ représentant un sommet de X :

- $V^+(G) = V_{\text{poly}} \cup V_{\text{prev}} \cup V_{\text{next}} \cup \{\tilde{u}, \tilde{v}\}$ avec :
  - V_{poly} l'ensemble des sommets-extrémités du polygone $\mathcal{P}$,
  - $V_{\text{next}} = \textsc{Next}(\hat{X} \cap \mathcal{B}) \setminus \hat{X}$, et $V_{\text{prev}} = \textsc{Next}^{-1}(\hat{X} \cap \mathcal{B}) \setminus \hat{X}$,
  - $\tilde{u}$ et $\tilde{v}$ deux sommets supplémentaires.
- $E^+(G) = E_{\text{poly}} \cup E_{\text{sch}} \cup E_{u-v} \cup E_{u\text{-next}} \cup E_{v\text{-next}} \cup E_{\text{next-next}}$ avec :
  - E_{poly} l'ensemble des paires x, y de $V_{\text{poly}} \cup V_{\text{prev}} \cup V_{\text{next}}$ telles qu'il n'existe pas de sommet $z \in \hat{X} \cup V_{\text{poly}} \cup V_{\text{prev}} \cup V_{\text{next}}$ satisfaisant $\textsc{Id}(x) < \textsc{Id}(z) < \textsc{Id}(y)$,
  - E_{sch} l'ensemble des paires x, y de $V_{\text{poly}} \cup V_{\text{prev}} \cup V_{\text{next}}$ telles que x et y sont associés par le couplage du schéma polygonal,
  - E_{u-v} est le singleton $\{(\tilde{u}, \tilde{v})\}$ s'il existe $\hat{u}_i, \hat{v}_j$, représentant u, v, qui sont connectés dans $\hat{G} \setminus \hat{X}$, et l'ensemble vide sinon,
  - $E_{u\text{-next}}$ est l'ensemble des paires $\tilde{u}, x$ telles qu'il existe un représentant $\hat{u}_i$ de u connecté à $x \in V_{\text{next}}$ dans $\hat{G} \setminus \hat{X}$,

– $E_{v\text{-next}}$ est l'ensemble des paires $\tilde{v},x$ telles qu'il existe un représentant $\hat{v}_i$ de v connecté à $x \in V_{\text{next}}$ dans $\hat{G} \setminus \hat{X}$,
– $E_{\text{next-next}}$ est l'ensemble des arêtes d'une forêt couvrante maximale du graphe G_{next}, graphe dont les sommets forment V_{next} et les arêtes les paires de sommets connectés de $\hat{G} \setminus \hat{X}$.

Lemme 2 *Etant donnée une requête (u,v,X), on peut construire le graphe G^+ à l'aide de l'algorithme A_0 du schéma $\mathcal{L}_0$, et à partir des étiquettes de u, v et de celles de X. D'après [3], le temps de construction est $O(g|X| \cdot \log(g|X|))$.*

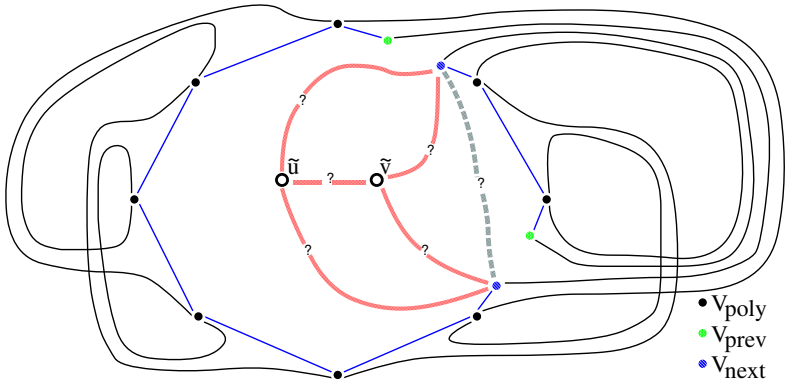

FIGURE 2: Graphe auxiliaire G^+ fonction de (u,v,X). Les arêtes marquées « ? » dépendent du schéma $\mathcal{L}_0$.

Lemme 3 *Répondre à la requête* $\textsc{Conn}(u,v,X)$ *est équivalent à tester la connexion de $\tilde{u}$ et $\tilde{v}$ dans G^+.*

3 Conclusion

D'après le lemme 3, la requête $\textsc{Conn}(u,v,X)$ peut être donc résolu en temps $O(g|X|)$, une fois le graphe G^+ construit. Cependant, le lemme 2 peut être raffiné en considérant le sous-graphe auxiliaire $G^+ \setminus \{\tilde{u},\tilde{v}\}$, ne dépendant que de X, à partir duquel il est possible de construire (en temps linéaire) une structure de données surportant les requêtes de connexité (u,v) en temps $O(\log(g|X|))$, pour tout $u,v \notin X$. En utilisant des structures de données transdichotomiques [2], il même est possible de réduire le temps de réponse à $O(\min\{\log(g|X|)/\log\log(g|X|), \sqrt{\log n/\log\log n}\})$ sans détériorer le temps de précalcul.

Références

[1] I. Abraham, S. Chechik, C. Gavoille, and D. Peleg. Forbidden-set distance labels for graphs of bounded doubling dimension. In 29^{th} *Annual ACM Symp. on Principles of Distributed Computing (PODC)*, pp. 192–200, 2010.

[2] T. M. Chan and M. Pătraşcu. Transdichotomous results in computational geometry, I : Point location in sublogarithmic time. *SIAM Journal on Computing*, 39(2) :703–729, 2009.

[3] B. Courcelle, C. Gavoille, M. M. Kanté, and D. A. Twigg. Connectivity check in 3-connected planar graphs with obstacles. In *International Conference on Topological & Geometric Graph Theory*, vol. 31, pp. 151–155. Electronic Notes in Discrete Mathematics, 2008.

[4] B. Courcelle and D. A. Twigg. Compact forbidden-set routing. In 24^{th} *Annual Symp. on Theoretical Aspects of Computer Science (STACS)*, vol. 4393 of Lecture Notes in Computer Science, pp. 37–48. Springer, 2007.

[5] D. Eppstein, Z. Galil, G. F. Italiano, and A. Nissenzweig. Sparsification – A technique for speeding up dynamic graph algorithms. *Journal of the ACM*, 44(5) :669–696, 1997.

[6] D. Eppstein, Z. Galil, G. F. Italiano, and T. H. Spencer. Separator-based sparsification II : Edge and vertex connectivity. *SIAM Journal on Computing*, 28(1) :341–381, 1998.

[7] M. Pătraşcu and M. Thorup. Planning for fast connectivity updates. In 48^{th} *Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, pp. 263–271. IEEE Computer Society Press, 2007.

[8] G. Vegter and C. K. Yap, Computational complexity of combinatorial surfaces. In 6^{th} *Annual ACM Symp. on Computational Geometry (SoCG)*, ACM-SIAM, pp. 102–111, 1990.